PSPICE tutorial: MOSFETs

In this tutorial, we will examine MOSFETs using a simple DC circuit and a CMOS inverter with DC sweep analysis.

This tutorial is written with the assumption that you know how to do all of the basic things in PSPICE: starting a project, adding parts to a circuit, wiring a circuit together, using probes, and setting up an using a simulation profile.

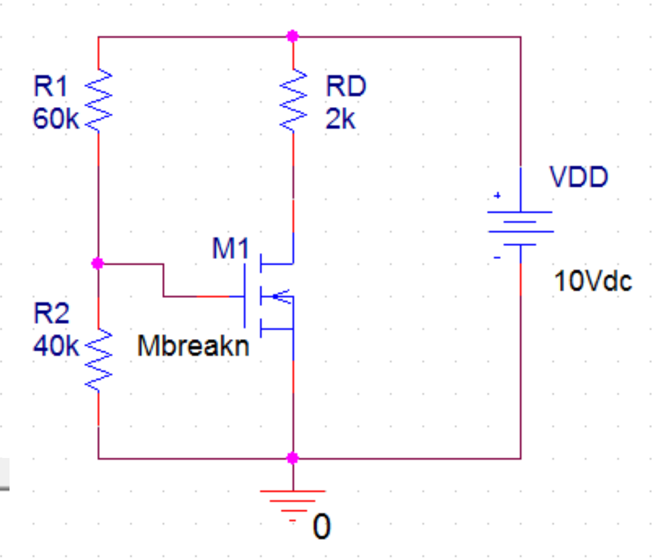
MOSFET models

Once again, we will use the device models from the Breakout library. As we have seen previously, we can easily change the parameters of these "bare-bones" models so that our circuits are identical to the academic exercises that we do in class.

The MOSFET models that we will use are the the MbreakN3 and MbreakN4 devices for NMOS and the MbreakP3 and MbreakP4 models for PMOS. These are nearly identical, with a subtle difference between the "3" and "4" versions. The "4" versions have 4 terminals (D, S, G + body) – the body connection must be wired up explicitly. In the "3" version, the body is already connected to the source and so there are only three connections (D, S, and G). As you might expect, an MbreakN4 can be turned into an MbreakN3 by connecting the body to the source when wiring up a circuit. (And the same for MbreakP.) In any situation where the transistor body is always at the same voltage as the source, we can use either model, with appropriate wiring. However, in situations where the source may be at a different potential from the body, we must use the 4-terminal model. An example would be in the use of pass transistors serving as switches. We will look at some of those situations in a future tutorial.

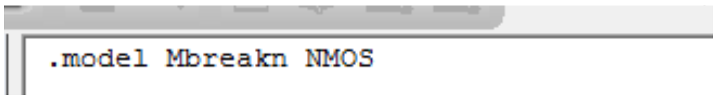
If the MbreakN and MbreakP models are used without any modification, PSPICE will use the default values of basic parameters: the threshold voltage will be 0 V and K will be equal to 0.1 mA/V^2. We will see how to change the parameters in the examples below.

Simple DC simulation



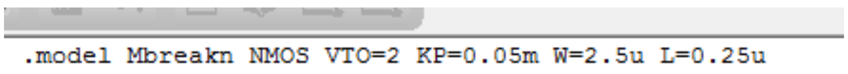
Build the circuit at right using the MbreakN3 model for the NMOS.

To change the parameters of the NMOS, click on it to highlight it. Then right-click on the highlights symbol and choose the "Edit PSPICE model…" item form the pop-up window. In the dialog that appears is a line of text that defines MbreakN as being a default NMOS.



```
.model Mbreakn NMOS
```

Change the basic parameters to $V_T = 2$ V and $K_n = 0.25$ mA/V^2 by adding text to the definition as shown at right.



```
.model Mbreakn NMOS VTO=2 KP=0.05m W=2.5u L=0.25u
```

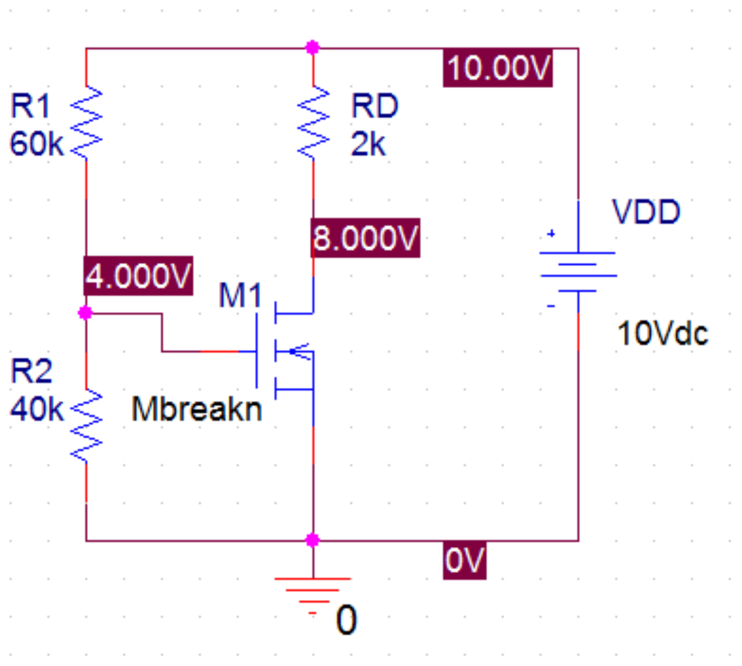
VTO ("oh" not "zero") is the threshold voltage. KP is the same as the product $\mu_n C_{ox}$ that we used in class. *W* is the gate width and *L* is the gate length. (Recall, that in PSPICE, u is used for μ.) So, the current parameter *K* (as used in class) for this transistor is

$$K_n = 0.5 \cdot KP \cdot (W/L) = 0.5(0.05 \text{ mA/V}^2)(2.5\ \mu\text{m})/(0.25\ \mu\text{m}) = 0.25 \text{ mA/V}^2.$$

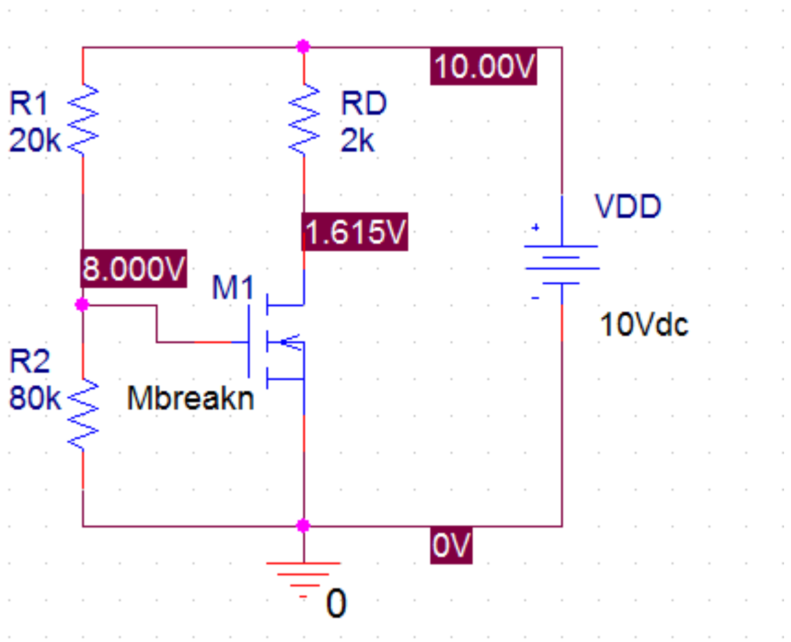
Since K_n is the product of several parameters, we could, in principle, choose any combination of those parameters that gives us the desired value of K_n. For example, we could have used KP =0.25 W=1 L=1 and the model would be identical. For our initial exercises with MOSFETs, it is important only to get the correct value of K_n. However, in the future, as our work with MOSFETs becomes more sophisticated, we will start to worry about things like parasitic capacitances in transient analyses. At that point, it will be important to have the correct gate sizes and the correct value for KP. We will also look at an alternative method to specify W and L for individual transistors within a circuit. But for now, we can pick any combination of KP, W, and L that gives that desired value for K_n.

Once the changes the MOSFET description have been made, we must save the description (Choose "Save…" from the file menu), and then we can close the dialog window.

Create a new simulation profile and choose the "Bias point" simulation option from the drop down menu. Run the simulation and display the DC voltages on the circuit by clicking on the "V" icon in the toolbar. The results are shown below. The NMOS is operating in saturation, and so it is easy to compute the voltages and currents by hand if we want to "check" PSPICE.

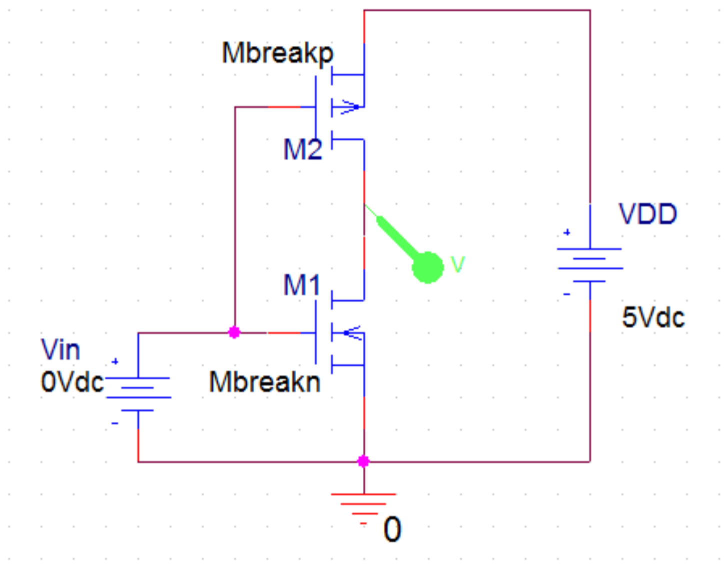


Changing the biasing resistors (hence changing the gate voltage) can push the NMOS into ohmic operation.

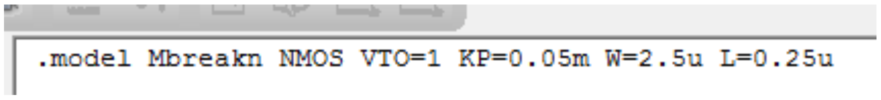


CMOS inverter

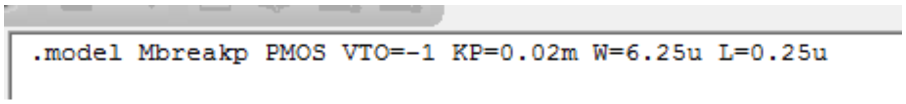
Add a PMOS transistor (MbreakP3 from the Breakout menu) to make a CMOS inverter, as shown below. (To turn the PMOS upside-down, use the “Mirror vertically” menu item from the right-click pop-up when placing the transistor.)



Edit the properties of the NMOS and PMOS to make them matched. ($K_n = K_p = 0.25$ mA/V^2) and $V_{Tp} = -V_{Tn} = 1$V). Again, we could choose any combination of KP, W, and L to obtain the desired K-values. However, we will choose numbers that are somewhat in line with what is seen in a real CMOS transistors, where $\mu_n \approx 2.5\mu_p$. (Holes are slower than electrons.)

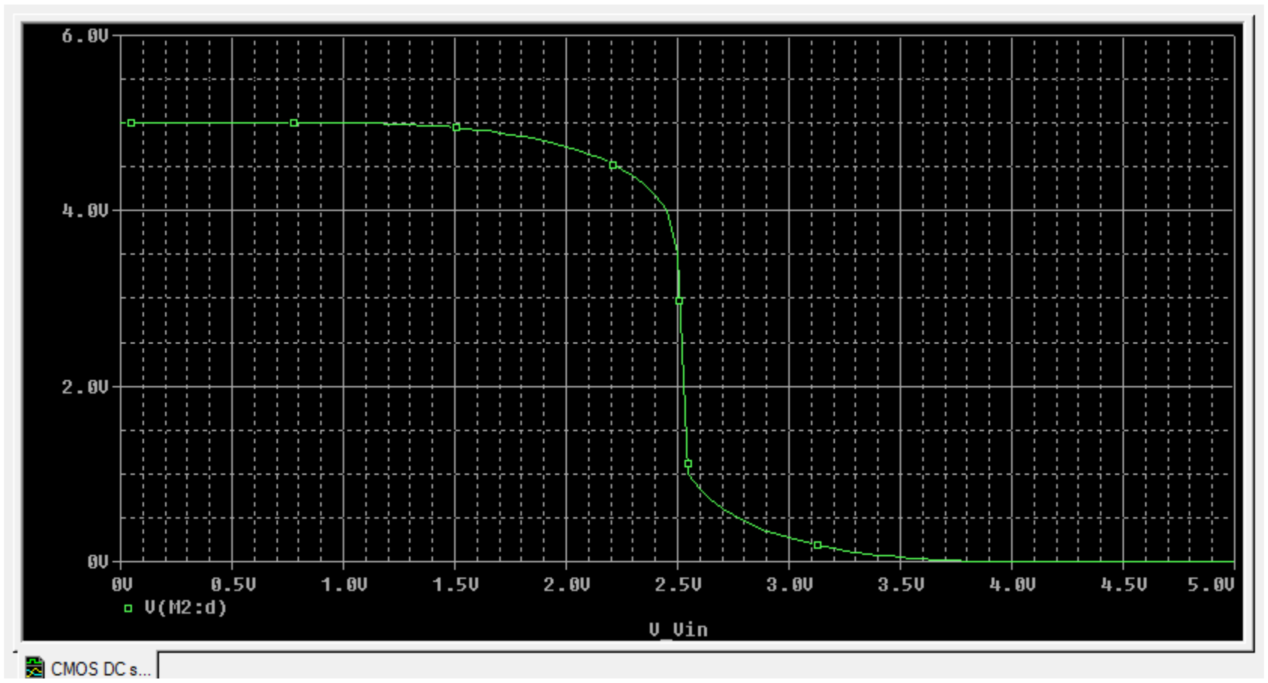
For the NMOS: 

```
.model Mbreakn NMOS VTO=1 KP=0.05m W=2.5u L=0.25u
```

For the PMOS: 

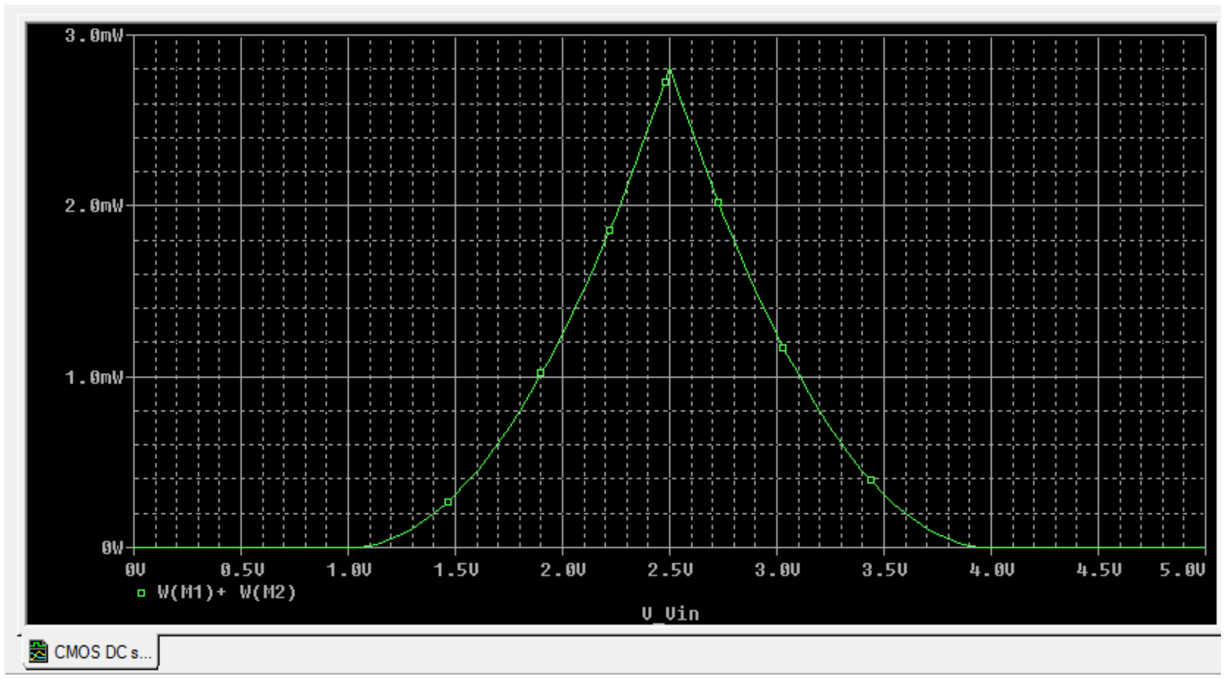
```
.model Mbreakp PMOS VTO=-1 KP=0.02m W=6.25u L=0.25u
```

Now set up a DC sweep simulation that sweeps Vin from 0 to 5 V in 0.05 V increments. (Review the DC Sweep tutorial for details on how to do this, if you've forgotten.) Run the simulation. If everything was set up correctly, the voltage transfer characteristic (VTC) should be plotted.



The VTC is symmetric, as expected when using matched transistors. Note: The sharp transition does not appear to be perfectly vertical. This is due to the fact that we are using a finite number of points in the DC Sweep. In the ideal case, the transition should be perfectly vertical, but only one point point out of the 101 points set up in the simulation will match up with that transition. The effect is to make the transition to appear to have a bit of a slope. We should point out that real inverters (made with real transistors) will have a bit of slope, because real transistors are not ideal.

CMOS inverters are all about reducing power. We can look at the static power as a function of input voltage. We could go back and add a power probe to the circuit. Or we can simple change the plot while we are in the "plot view". Go to the trace menu and choose "Remove all traces". Then choose the the trace menu item "Add trace…". In the dialog window that opens, enter W(M1)+W(M2) in the expression box at the bottom. (Or choose those items from the laundry list that is shown.) This will plot the total power dissipated in the NMOS and PMOS together. The resulting plot is shown below.

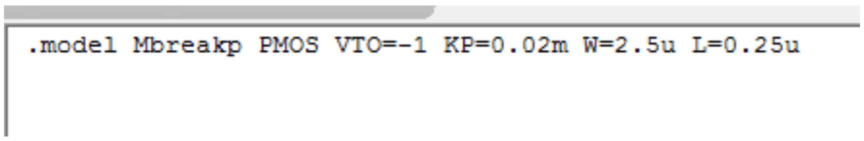


The plot confirms what we already knew – there is no power dissipated when the inverter is in its high or low output state. Power is only used when transitioning from one state to the other.

Editorial comment on notation: In class, we are using K_P to denote the total current parameter for a PMOS transistor. SPICE uses KP to denote μC_{ox} – the mobility-capacitance product for either NMOS or PMOS transistors. We should take care not to become confused about which is which.

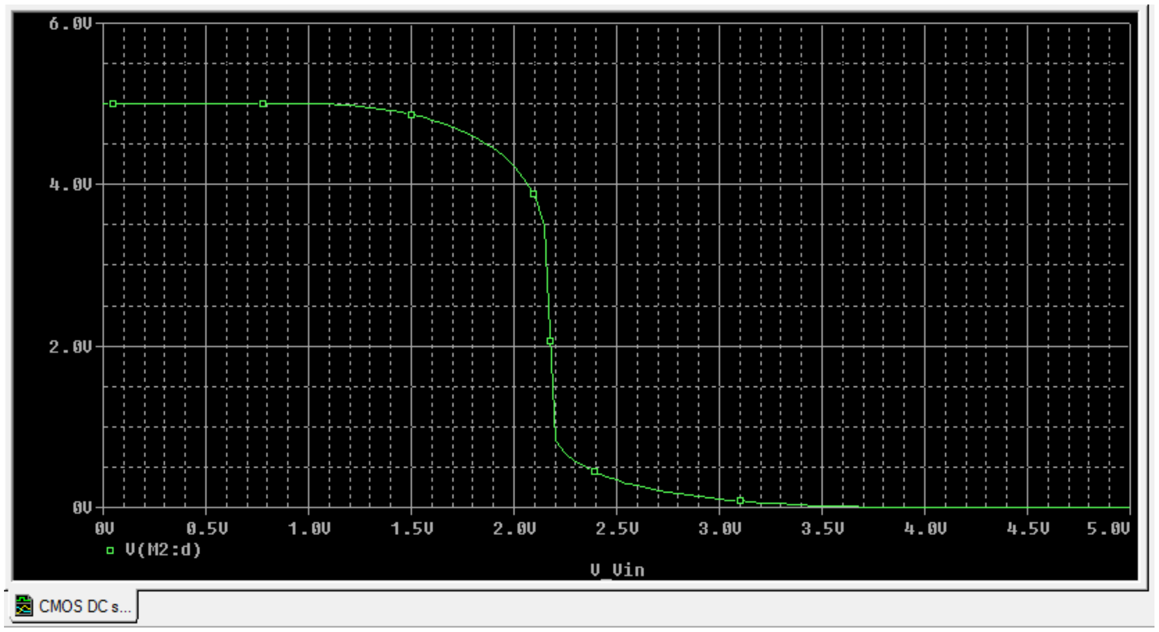
Finally, let's make the inverter unmatched by making the NMOS and PMOS have exactly the same size. This has the advantage of making the entire inverter smaller but will certainly have an effect on other aspects of inverter performance.

Change the PMOS description to:



```
.model Mbreakp PMOS VTO=-1 KP=0.02m W=2.5u L=0.25u
```

Now $K_n \approx 2.5K_p$. Re-running the simulation with these new numbers gives the VTC shown below.



The mis-match in the transistors shifts the transition point by about 0.3 V. This may not have have a huge effect on noise margins, and might appear to be a reasonable trade-off (smaller physical size for reduced noise margins). However, the mismatch will also affect switching speeds, and we should investigate that issue before declaring the smaller, un-matched inverter to be a winner. We will leave that investigation for a future tutorial or homework problem.